PATENT CLAIMS

1. Method for processing data, characterized in that a Petri net is encoded, written into a memory and read and performed from that memory by at least one instance, wherein transitions of the Petri net read from at least one tape and/or write on at least one tape symbols or symbol strings, with the aid of at least one head.

2. Method according to claim 1, characterized in that the Petri net, the head or the heads and the tape or the tapes form a universal Turing machine.

3. Method according to claim 1 or claim 2, characterized in that at least one second Petri net, in particular encoded with the properties of the Petri net described in claim 1, is written into a memory and is read and executed from this memory by at least one instance, wherein transitions of each Petri net can send symbols or symbol strings via at least one optionally existing channel, which can be received by transitions of other Petri nets via this channel or these channels.

4. Method according to one of claims 1 to 3, characterized in that a Petri net has access to a marker- or state transition table, respectively, and optionally to at least one output table or a combination of both, and by doing so determines a derived marker or a derived state, respectively, and optionally at least one output, depending from the marker and the state, respectively, and optionally depending from an optionally existing input.

5. Method for performing the method according to claim 4, characterized in that the switching of the transitions of a Petri net is performed by a processor, wherein the processor has at least one processor instruction, which processes the marker- or state

.../29

transition table, respectively, and optionally at least one output table or a combination of both as the operands.

6. Method according to claim 3, characterized in that a co-operation of Petri nets constitutes a Turing machine.

7. Method according to one of claims 3 and 6, characterized in that at least a part of a program is translated into a Petri net or a co-operation of Petri nets.

8. Method according to any one of claims 3 to 7, characterized in that the Petri nets are executed by a composition instruction, wherein a third Petri net, equivalent to the co-operating first and second Petri nets with respect to the external input/output behaviour, except output delays, is constituted with the aid of the first and second petri net.

9. Method for processing data, except public key encryption methods based on the composition of finite automates, said method being in connection with one of claims 1 to 8 in particular, and characterized in that data-processing, co-operating nets are composed, the composition result is encoded, written into a memory and read and executed from this memory by at least one instance, wherein the composition result is a net which is equivalent to its components with respect to the external input/output behaviour, except output delays.

10. Method according to any one of claims 1 to 8 and to claim 9, characterized in that the components and the composition result are Petri nets, wherein the transitions of the components can receive and send symbols or symbol strings via optionally existing channels.

.../30

11. Method according to claim 8 or 10, characterized in that the Petri nets constitute sequential machines M_Ω with optionally plural input channels and optionally plural output channels, C is a finite set of channels, Δ is a finite set of finite alphabets, γ: $C \rightarrow \Delta$, $\Omega = (C, \Delta, \gamma)$ is a communication rule,

$$E_\Omega = \{e | e = \{(c,\sigma) | \sigma \in \gamma(c) \wedge ((c,\sigma_1) \in e \wedge (c,\sigma_2) \in e \Rightarrow \sigma_1 = \sigma_2)\}\} \cup \{\emptyset\}$$

is a set of input/output events and S is a finite set of states and

$$M_\Omega := \{(S, E_\Omega, \delta, \beta, s_0) | \delta : R \rightarrow S \wedge \beta : R \rightarrow E_\Omega \wedge R \subset S \times E_\Omega$$
$$\wedge (\forall[(s,x),y] \in \beta \forall (c_x,\sigma_x) \in x \forall (c_y,\sigma_y) \in y : c_x \neq c_y) \wedge s_0 \in S\},$$

B with $B \subseteq C$ is a set of internal synchronization channels and the composition $comp_B : M_\Omega^n \rightarrow 2^{M\Omega}$ of sequential machines is defined as

.../31

$$
\begin{aligned}
&comp_B := \Big\{ \Big((K_1, \dots, K_n), \widetilde{K}\Big) \;\Big| \\
&\quad (K_1, \dots, K_n) = ((S_1, E_\Omega, \delta_1, \beta_1, s_{0_1}), \dots, (S_n, E_\Omega, \delta_n, \beta_n, s_{0_n})) \\
&\quad \wedge \exists T = \{\; ((x_1, \dots, x_n), (y_1, \dots, y_n), (s'_1, \dots, s'_n), \widetilde{x}, \widetilde{y}) \mid \\
&\qquad ([(s_{0_1}, x_1), s'_1], \dots, [(s_{0_n}, x_n), s'_n]) \in \delta_1 \times \dots \times \delta_n \\
&\qquad \wedge ([(s_{0_1}, x_1), y_1], \dots, [(s_{0_n}, x_n), y_n]) \in \beta_1 \times \dots \times \beta_n \\
&\qquad \wedge \exists H_x = \bigcup_{i \in \{1, \dots, n\}} x_i \; \exists H_y = \bigcup_{i \in \{1, \dots, n\}} \beta_i(x_i) : \\
&\qquad H_x \in E_\Omega \;\wedge\; H_y \in E_\Omega \\
&\qquad \wedge \forall (c, \sigma) : \; (c \in B \Leftrightarrow (c, \sigma) \in H_x \cap H_y) \\
&\qquad \wedge \widetilde{x} = H_x \setminus H_y \;\wedge\; \widetilde{y} = H_y \setminus H_x \;\} \\
&\quad \exists \widetilde{M}'_\Omega = \{\; \widetilde{K}' \mid \exists ((x_1, \dots, x_n), (y_1, \dots, y_n), (s'_1, \dots, s'_n), \widetilde{x}, \widetilde{y}) \in T : \\
&\qquad \widetilde{K}' = comp_B ([(S_1, E_\Omega, \delta_1, \beta_1, s'_1), \dots, (S_n, E_\Omega, \delta_n, \beta_n, s'_n)]) \;\} : \\
&\quad \widetilde{K} = \Big(\widetilde{S}, E_\Omega, \widetilde{\delta}, \widetilde{\beta}, \widetilde{s_0}\Big) \\
&\qquad \wedge \widetilde{S} = (s_{0_1}, \dots, s_{0_n}) \cup \bigcup_{\left(\widetilde{S}', E'_\Omega, \widetilde{\delta}', \widetilde{\beta}', \widetilde{s}'_0\right) \in \widetilde{M}'_\Omega} \widetilde{S}' \\
&\qquad \wedge \widetilde{\delta} = \{\; [((s_{0_1}, \dots, s_{0_n}), \widetilde{x}), (s'_1, \dots, s'_n)] \mid \\
&\qquad\quad ((x_1, \dots, x_n), (y_1, \dots, y_n), (s'_1, \dots, s'_n), \widetilde{x}, \widetilde{y}) \in T \;\} \\
&\qquad\quad \cup \bigcup_{\left(\widetilde{S}', E'_\Omega, \widetilde{\delta}', \widetilde{\beta}', \widetilde{s}'_0\right) \in \widetilde{M}'_\Omega} \widetilde{\delta}' \\
&\qquad \wedge \widetilde{\beta} = \{\; [((s_{0_1}, \dots, s_{0_n}), \widetilde{x}), \widetilde{y}] \mid \\
&\qquad\quad ((x_1, \dots, x_n), (y_1, \dots, y_n), (s'_1, \dots, s'_n), \widetilde{x}, \widetilde{y}) \in T \;\} \\
&\qquad\quad \cup \bigcup_{\left(\widetilde{S}', E'_\Omega, \widetilde{\delta}', \widetilde{\beta}', \widetilde{s}'_0\right) \in \widetilde{M}'_\Omega} \widetilde{\beta}' \\
&\qquad \wedge \widetilde{s_0} = (s_{0_1}, \dots, s_{0_n}) \;\Big\}.
\end{aligned}
$$

12. Method according to claim 9, characterized in that the data-processing nets are constituted by a translation of algorithms.

.../32

13. Method according to any one of claims 9 to 12, characterized in that at least one component is a cryptological component.

14. Method according to claim 13, characterized in that at least one component deflates compressed data.

15. Method according to one of claims 13 or 14, characterized in that one component reads data and adds a feature or watermark to the data by changing these data, which is not or only slightly obstructing the use of these data.

16. Method according to claim 13, characterized in that an encoder and a combiner of plural input channels are composed into one output channel, wherein the encoder and the combiner are Petri nets, the combiner maps the data received via the input channels on the output channel, and the output of the combiner is the input for the encoder.

17. Method according to claim 13, characterized in that a decoder and a separator are combined, wherein the decoder and the separator, respectively, is a Petri net and may be an inversion of an encoder and a combiner, respectively, which has been composed with a combiner or encoder, respectively, according to the method described in claim 16, the separator maps the data received via the input channel on the output channel, and the output of the encoder is the input for the separator.

18. Method according to any one of claims 13 to 17, characterized in that at least one component is a cryptological component which receives and processes data from a cryptological function which is executed in a protected manner, wherein the composition result does not work or works in an erroneous manner in the case that no data or erroneous data are received from the cryptological function.

.../33

19. Method according to claim 18, characterized in that a composition is executed omitting the cryptological component or at least one of the cryptological components, wherein the composition result has at least one limitation with respect to usability, compared with the composition result which has been formed without said omission.

20. Method for processing data, in particular in connection with one of claims 1 to 19, characterized in that a data-processing net or a program, respectively, receives and processes second data from a cryptological function which is executed in a protected manner, and the data-processing net or the program, respectively, does not work or works in an erroneous manner in the case that no second data or erroneous second data are received, wherein the cryptological function is fixedly attached to the device which executes the data-processing net or the program, respectively.

21. Method according to claim 20, characterized in that a value exceeding the calculation of a function value of the cryptological function is stored such that it is not readable or changeable for an aggressor, and on further calculation of a further function value this value influences the result of the further calculation, the value changing according to a predetermined rule.

.../34